accusoft®

Jump Start Your PrizmDoc Viewer Development with Docker

Running containers instead of virtual machines is becoming more and more popular. This is because Docker significantly simplifies deployment, reduces infrastructure requirements, and is very effective from a resource consumption point of view.

If you want to add document viewing capabilities to your web application and benefit from the improvements Docker provides, then PrizmDoc Viewer Docker images are a great choice.

In a matter of minutes, you can get a local PrizmDoc Viewer demo working, open your documents, play with the viewer, and even proceed with prototyping your application code. As you move into development and then deployment, you will also enjoy the ease of both setting up and starting up PrizmDoc containers.

Continue reading to learn more about how PrizmDoc Viewer Docker can streamline your development and deployment processes, see what it takes to get started, and get some useful hints on working with PrizmDoc Viewer Docker.

What You Get with PrizmDoc Viewer Docker

There are a lot of reasons why PrizmDoc Viewer Docker is a great choice, not just for evaluation, but also for use in development and production.

Seamless Installation

You don't have to worry about specific OS requirements, conflicts with other software installed on the server, or differences between your development and deployment environments.

Fast Startup

Starting up PrizmDoc containers only takes a matter of seconds. You don't need to create and keep an image with pre-installed PrizmDoc to be able to start up a fresh instance quickly.

Scaling and Recycling

In the real world, users usually sleep at night, and work with documents during daytime office hours. Even if your customers are healthcare organizations working around the clock, or are distributed across multiple time zones, the load on PrizmDoc servers will vary throughout the day and throughout the week. To reduce computing costs, you will likely start up PrizmDoc instances when the load increases, and recycle them when the load decreases. Using Docker-based PrizmDoc Viewer allows you to start up instances quickly and easily.

What You Get with PrizmDoc Viewer Docker

Easy Recovery

Although PrizmDoc services have internal machinery for keeping themselves up and productive, there can still be cases when things go wrong. If a PrizmDoc instance reports itself as unhealthy, it is best to simply delete it and start a new one, which is also easiest using Docker.

Since the production version of prizmdoc-server keeps its cache on the host file system, the cache will not be lost when the unhealthy container is deleted, and can be picked up by the fresh container.

Preview Images

Here at Accusoft, we stay in touch with customers, whether we are working on new features or fixing bugs. With Docker, you can easily pull and run preview versions of PrizmDoc Viewer images to try our new features and fixes.

Special Case: Rendering MS Office Files Natively

All kinds of documents that are supported by PrizmDoc, including Microsoft Office documents, are also supported by PrizmDoc Docker. However, in some cases you may need MS Office documents to render natively, using the MS Office engine. In that case, you will need to use a regular PrizmDoc Windows installation for MS Office conversions. You can still benefit from using PrizmDoc Docker for the rest of your conversions, and set up MSO connectivity specifically for rendering Office documents.

Quick Start

To get started with Dockerized PrizmDoc Viewer, we suggest you try the accusoft/prizmdoc-viewer-eval image. It combines the complete PrizmDoc backend and a demo web application in a single image.

All you need for this is a Linux, Windows, or Mac system with at least 2 CPUs, 8 GB of RAM, and Docker installed. Note, if you are using Docker Desktop for Windows, make sure it is switched to Linux containers.

Open your terminal and execute just two commands to run this evaluation container:

1. Pull the latest image:

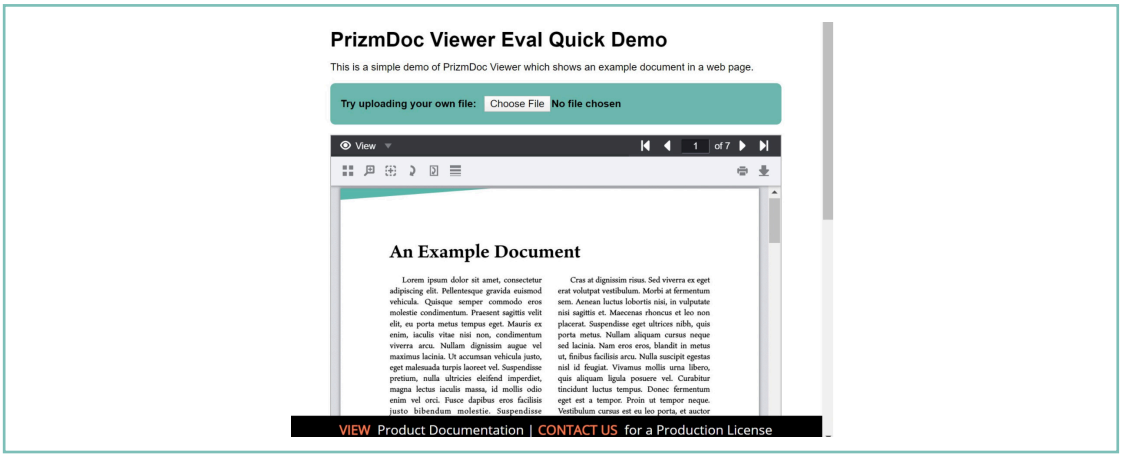
```
docker pull accusoft/prizmdoc-viewer-eval:latest
```

2. Run container:

```
docker run --rm -p 8888:8888 -p 3000:3000 -p 18681:18681 -e
ACCEPT_EULA=YES --name prizmdoc-viewer-eval
accusoft/prizmdoc-viewer-eval:latest
```

Please note that the environment variable ACCEPT_EULA=YES indicates that you have accepted the PrizmDoc Viewer license agreement.

Once you see "Application running at http://localhost:8888" in the console, open this URL in the browser to see the demo page:



The demo is running in evaluation mode with a fixed-feature set. Please refer to the accusoft/prizmdoc-viewer-eval DockerHub page for complete instructions on how to unlock all PrizmDoc Viewer features for in-depth evaluation. Please note that the accusoft/prizmdoc-viewer-eval image is suitable for evaluation only, and you will need to use production images for development and deployment.

Developing Your Application

Although prizmdoc-viewer-eval contains a complete PrizmDoc backend and can be used for product evaluation, you will need to switch to the production PrizmDoc images for developing your application.

The key difference is that production images allow you to adjust configuration settings, such as cache lifetime or display of gridlines in Excel documents. They also let you keep PrizmDoc cache folders and logs outside of the container, so they remain available in case you delete or recycle the container.

The production version of PrizmDoc Viewer also gives you more flexibility in server instance scaling because of its two-component structure, discussed below.

Getting Started with Production Images

The production version of PrizmDoc comes as two images:

• prizmdoc-application-services, also referred to as PAS, provides the high-level REST API for your web application. Along with viewing functionality, it takes care of storing annotations and pre-converted content for viewing.

• prizmdoc-server is the technical heart of the product, where the actual document conversions take place.

Developing Your Application

If you use PrizmDoc Viewer in a clustered environment, you can track the load on each kind of PrizmDoc instance, and scale either the number of PAS or PrizmDoc Server containers. For example, if the number of users is relatively small, but the documents are complex, you will likely have more PrizmDoc Server instances than PAS instances.

However, running PAS and PrizmDoc Server on the same instance is also fine, and quite likely this is the option you will choose for your local development.

PrizmDoc Viewer containers expect configuration files to be available on your server. Running containers in “init-config” mode will create default config files. You will need to update prizmdoc-application-services config with the address of your prizmdoc-server instance. If you already have a license for PrizmDoc, you would want to add it to your prizmdoc-server config to unlock evaluation mode and have access to all PrizmDoc Viewer features. You only need to run this step once. See prizmdoc-server and prizmdoc-application-services readmes for details.

One-Click Startup with Docker Compose

As we mentioned above, when you develop or deploy your PrizmDoc Viewer application, you need to deal with two Docker images (PAS and PrizmDoc Server). The easiest way to spin up an entire PrizmDoc Viewer backend on a single machine is to use Docker Compose.

To begin, create a docker-compose.yml file with the following contents:

```
version: “3.7”
services:
  prizmdoc-server:
    image: accusoft/prizmdoc-server
    environment:
      ACCEPT_EULA: ‘YES’
    ports:
      - “18681:18681”
    volumes:
      - ./prizmdoc-server/config:/config
      - ./prizmdoc-server/logs:/logs
      #- ./prizmdoc-server/data:/data # Linux only
  pas:
    image: accusoft/prizmdoc-application-services
    environment:
      ACCEPT_EULA: ‘YES’
    ports:
      - “3000:3000”
    volumes:
      - ./pas/config:/config
      - ./pas/logs:/logs
      - ./pas/data:/data
```

This file describes containers to use, their names, exposed ports, and mapped folders.

This file describes containers to use, their names, exposed ports, and mapped folders.

Before you can actually start PAS and PrizmDoc Server, you're going to need config files for each of them. Run the following commands in the terminal:

```
docker-compose run --rm prizmdoc-server init-config

docker-compose run --rm pas init-config
```

This will create two config files:

```
• ./prizmdoc-server/config/prizm-services-config.yml
• ./pas/config/pcc.nix.yml
```

Next, you will need to customize the contents of the pcc.nix.yml file for PAS:

• **Important:** Change `pccServer.hostName` to `prizmdoc-server:`

```
pccServer.hostName: prizmdoc-server
```

When running multiple containers with Docker Compose, the container name is the hostname to use. So, for the pas container to be able to talk to the *prizmdoc-server container*, you need to change localhost to the name you are using use for the container: `prizmdoc-server`

• Choose your own secretKey value. This step is an optional security best practice.

You can find more information about configuring PAS here.

Now that you have prepared the config files, you're ready to actually start PAS and PrizmDoc Server:

```
docker-compose up
```

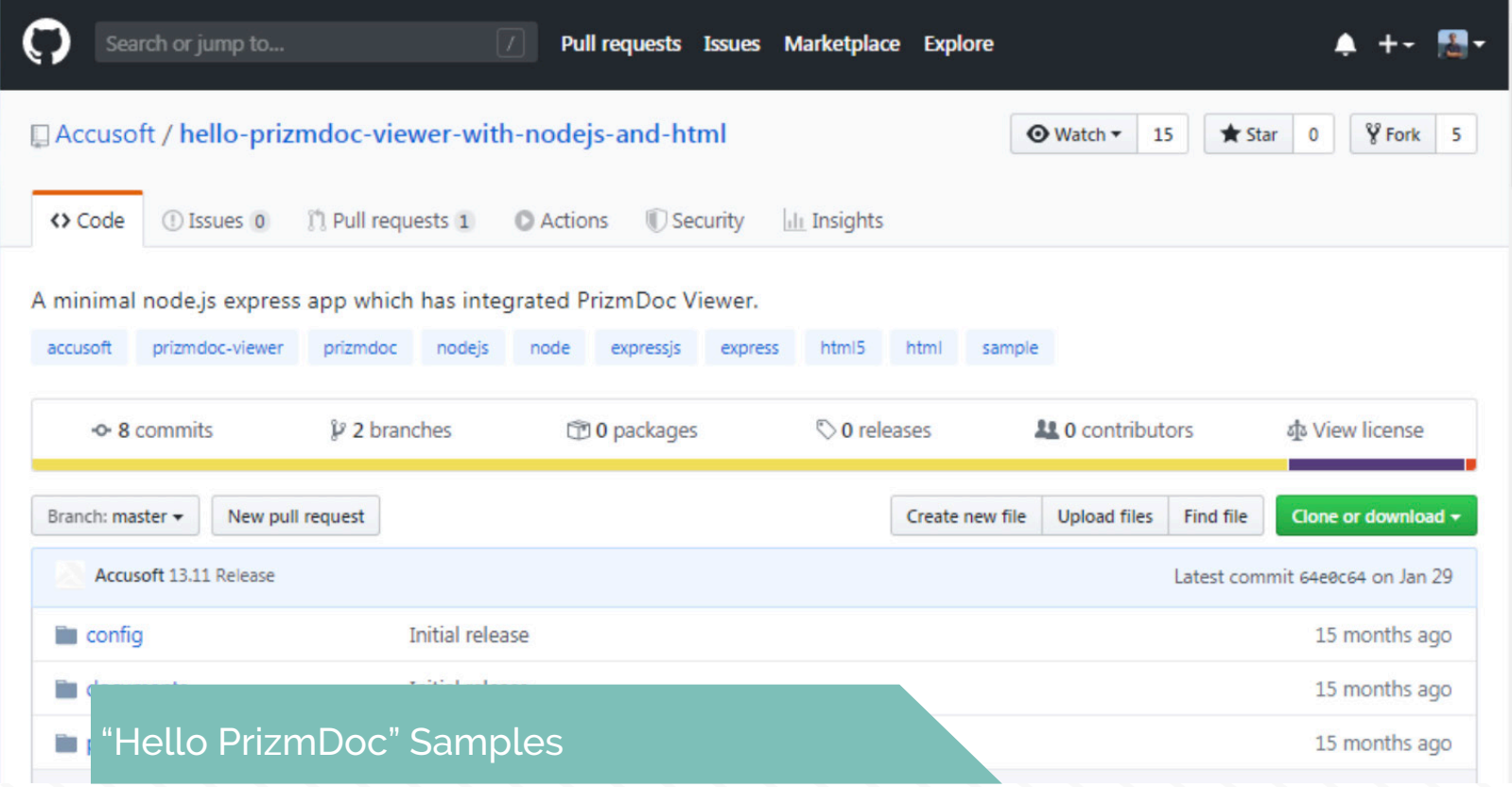
This should start containers for both PAS and PrizmDoc Server. You should see output like this:

```
Recreating wat_prizmdoc-server_1 ... done
Recreating wat_pas_1             ... done
Attaching to wat_prizmdoc-server_1, wat_pas_1
pas_1              | Starting pas
prizmdoc-server_1  | [info] Starting Prizm Content Connect Information Services…
….
prizmdoc-server_1  | [info] product-runner has been started correctly.
prizmdoc-server_1  | [info] Starting PCCIS Watchdog process...
prizmdoc-server_1  | [info] PCCIS Watchdog has been started correctly.
```

Wait until you see the following line before proceeding to the next step:

```
prizmdoc-server_1  | [info] PCCIS Watchdog has been started correctly.
```

The containers are running in the foreground so that you can see their startup output; you won't be able to issue further commands in this terminal.



"Hello PrizmDoc" Samples

The best way to start your web application development is to clone and play with one of our GitHub "Hello PrizmDoc" samples, such as hello-prizmdoc-viewer-with-nodejs-and-html, hello-prizmdoc-viewer-with-dotnet-and-html, or hello-prizmdoc-viewer-with-java-and-html: (view above)

You will need to set the sample application's PAS base URL config parameter to the public port exposed by your prizmdoc-application-services container, as described in the sample's readme file.

Build Your Container on Top of PrizmDoc

You may want to install additional components, such as fonts, on your PrizmDoc server. You can do this by creating your Docker image on top of PrizmDoc:

For example, to create an image with additional Korean language support, prepare a Dockerfile with the following contents:

```
FROM accusoft/prizmdoc-server:13.11
RUN apt-get update && apt-get install -y \
      language-pack-ko korean* \
      && rm -rf /var/lib/apt/lists/*
```

Then build the image:

```
docker build --tag prizmdoc-server-korean:13.11 .
```

Now you can use the `prizmdoc-server-korean:13.11` image.

Where to Go from Here

For further details, see:

PrizmDoc Viewer Home Page at accusoft.com
PrizmDoc Viewer Documentation

About the Author

Dmitry Shubin
Software Engineer

Dmitry Shubin graduated from the Moscow State University with a Bachelor's degree in Computer Science and a focus in computer graphics. He joined Accusoft in 1996 as a software developer for the ImageGear product and contributed in many areas including core design, graphic file formats support, and image processing. Dmitry is currently working on PrizmDoc Viewer. While mostly contributing to the product development, Dmitry is also passionate in sharpening the test suites. Away from work, Dmitry enjoys travelling, hiking, and playing his guitar around a campfire.

Ilya Shestakov
Software Engineer

Ilya Shestakov began his Accusoft career in 2014 as an engineer on the PrizmDoc Viewer team. With a Bachelor's of Science in Applied Mathematics, Ilya has contributed to the development of both backend microservices and front-end. In his spare time, Ilya enjoys running, snowboarding, and playing board games with his family.

About Accusoft

Accusoft is a software development company specializing in content processing, conversion, and automation solutions. From out-of-the-box and configurable applications to APIs built for developers, we help organizations solve their most complex content workflow challenges. Our patented solutions enable users to gain insight from content in any format, on any device with greater efficiency, flexibility, and security.